

# How Can We Conduct “Fair and Consistent” Hardware Evaluation for SHA-3 Candidate?

Shin'ichiro Matsuo<sup>1</sup> Miroslav Knežević<sup>2</sup> Patrick Schaumont<sup>3</sup> Ingrid  
Verbauwhede<sup>2</sup> Akashi Satoh<sup>4</sup> Kazuo Sakiyama<sup>5</sup> Kazuo Ota<sup>5</sup>

<sup>1</sup>NICT

<sup>2</sup>Katholieke Universiteit Leuven

<sup>3</sup>Virginia Tech

<sup>4</sup>AIST

<sup>5</sup>The University of Electro-Communications

NIST 2nd SHA-3 Candidate Conference

# Outline of This Talk

- Background of Hardware Evaluation
- Analysis of Hash Usages
- A Project for Fair and Consistent Evaluation
  - Outline of the Project
  - Setting of Hardware Evaluation
  - Evaluation Results
- What We Learned
- Conclusion

# Background

- Hardware evaluation plays important role in selecting SHA-3 algorithm.
- Several ongoing projects on hardware evaluation
- Validity and consistency are not well discussed.
- For “Fair” comparison, we must fix
  - Evaluation environment (platform)
  - Implementation method (design strategy)
  - Performance comparison method (evaluation criteria)
- Performance evaluation is complicated problem.
  - Design space
  - Constraints (area, throughput and energy)
  - Power consumption

# Analysis of Hash Usages

Hash Functions are widely used in ISO standards of cryptographic primitives,

- Public key encryption
- MAC
- Digital signature
- Authenticated encryption

In cryptographic protocols,

- Establishing secure channels
- Entity authentication
- Key exchange
- PKI
- Time-stamping
- Secure e-mail
- OTP

We must consider hash usages which is executed in hardware

- Hardware Security Modules (HSM):  
e.g. Issuing public key certificate in CA
- Smartcards:  
e.g. authentication and digital signature

# Fair and Consistent Evaluation

For complete comparison:

- Speed performance
- Size of Circuit
- Power Consumption

To do such comparison, we must fix

- Architecture of Hardware Implementation
- Common Platform (FPGA/ASIC)
- Interface Specification of Measurement

For fairness:

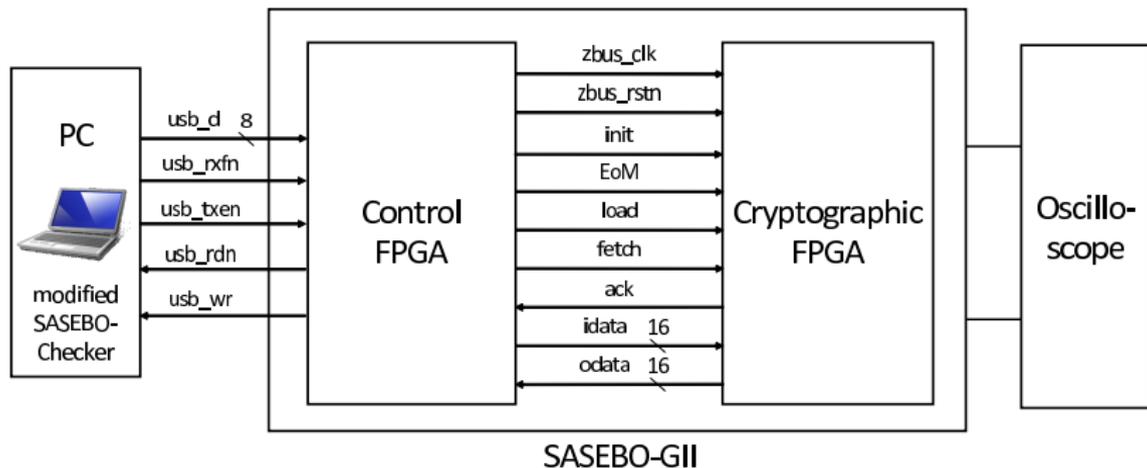
- Evaluation Environment is Open and Available to All Designers and Verifiers
- Unified and Common for All Candidates
- Results are Reproducible and Open for Public Verification

# Outline of the Evaluation Project

- World-wide collaboration
  - Share the workload for many number of 2nd-round candidates
  - More reliable than evaluation by single institute
  - Conducted by 5 different research teams (Japan, US and Belgium)
- Using common and publicly available evaluation platform
- Open RTL codes over the Internet for public verifiability (open-source like management)

# Evaluation Platform

Use SASEBO-GII FPGA Board



- 2 FPGAs

- Cryptographic FPGA (Xilinx Virtex-5 (xc5vlx30-3ff324))
- Control FPGA
- PC is connected via USB interface

# Defining Interface

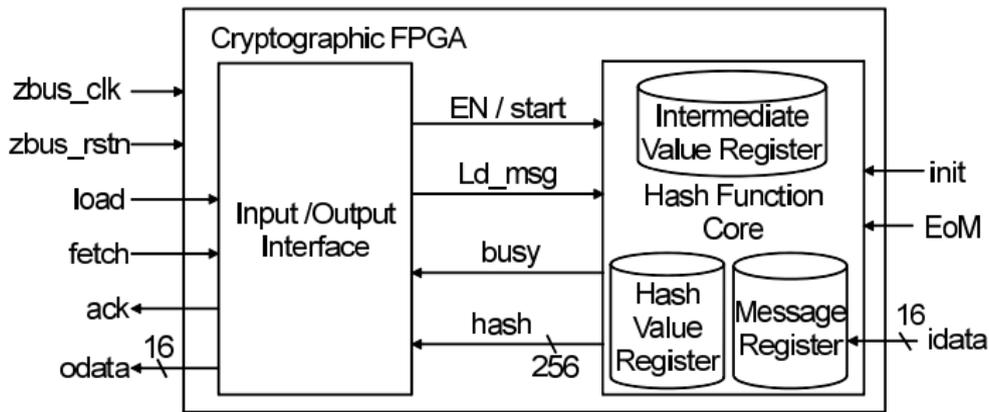
- Emulates API of Software Implementation
- Also define
  - Handshake Protocol: Use slave protocol
  - Wordlength: Must define to adapt to hardware specification. Use 16-bits interface
  - Control: Use additional control signals on the interface
  - Padding: Word-level padding

# Design Strategy

There are three types of architecture

- Fully Autonomous
- External Memory
- Core Functionality

We use Fully Autonomous architecture: suitable for integration into other architecture such as system-on-chip.



# Evaluation Items

- Speed Performance
  - Throughput for long message
  - Latency for short message
- Implementation Cost
  - Number of slices
  - registers
  - LUTs
- Power Consumption and energy cost

	Long Message (Throughput)	Short Message (Latency)
Interface + Core	$\frac{B \cdot f_{max}}{I_{in} + I_{core}}$	$\frac{M_p}{Th}$
Core Function Block	$\frac{B \cdot f_{max}}{I_{core}}$	$\frac{M_p}{Th_{core}}$

# Evaluation Results (Speed Performance)

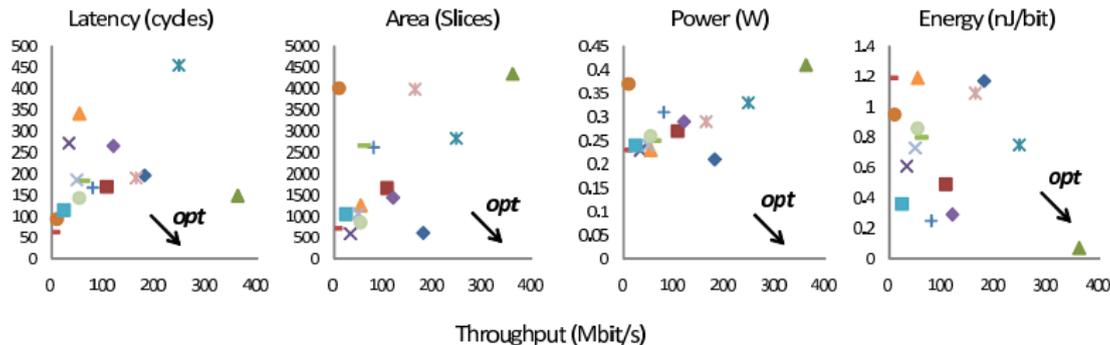
Implementation	Input Block Size [bits]	Max. Clock Freq [MHz]	Total Number of Clock Cycles [cycles]		Long Message Throughput [Mbps]	Short Message $M=1,024$ Latency [ $\mu$ s]
			I/F + Core	Core		
			SHA-256	512	260	148 (196)
BLAKE-32	512	115	121 (169)	22 (22)	487 (2,676)	3.57 (0.574)
BMW-256	512	34	98 (148)	2 (4)	178 (8,704)	10.12 (0.235)
CubeHash16/32-256	256	185	64 (272)	16 (176)	740 (2,960)	2.85 (1.297)
ECHO-256	1,536	149	407 (455)	99 (99)	562 (2,312)	3.05 (0.664)
Fugue-256	32	78	8 (93)	2 (39)	312 (1,248)	4.47 (1.321)
Grøstl-256	512	154	106 (164)	10 (20)	744 (7,885)	2.44 (0.260)
Hamsi-256	32	210	10 (63)	4 (9)	672 (1,680)	1.92 (0.690)
JH-256	512	201	135 (183)	39 (39)	762 (2,639)	2.25 (0.582)
Keccak(-256)	1,024	205	217 (265)	25 (25)	967 (8,397)	2.35 (0.244)
Luffa-256	256	261	57 (114)	9 (18)	1,172 (7,424)	1.31 (0.207)
Shabal-256	512	228	143 (341)	50 (200)	816 (2,335)	2.75 (1.316)
SHAvite-3 <sub>256</sub>	512	251	134 (185)	38 (38)	959 (3,382)	1.79 (0.454)
SIMD-256	512	75	142 (190)	46 (46)	270 (835)	6.32 (1.840)
Skein-256-256	256	115	75 (143)	21 (41)	393 (1,402)	3.20 (0.904)

# Evaluation Results (Size of Circuit and Power Consumption)

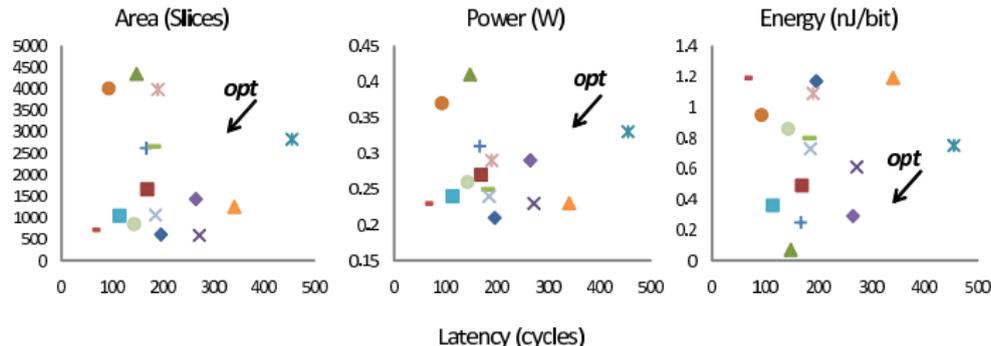
Implementation	Number of Occupied Slices	Number of Slice Registers	Number of Slice LUTs	Power [W]		Energy [ <i>nJ/bit</i> ]	
				Long Msg	Short Msg	Long Msg	Short Msg
SHA-256	609	1,224	2,045	0.21	0.21	1.17	1.17
BLAKE-32	1,660	1,393	5,154	0.27	0.27	0.49	0.49
BMW-256	4,350	1,317	15,012	0.41	0.41	0.07	0.09
CubeHash16/32-256	590	1,316	2,182	0.23	0.23	0.61	1.82
ECHO-256	2,827	4,198	9,885	0.33	0.33	0.89	0.89
Fugue-256	4,013	1,043	13,255	0.36	0.37	0.95	1.47
Grøstl-256	2,616	1,570	10,088	0.31	0.31	0.25	0.25
Hamsi-256	718	841	2,499	0.23	0.23	1.19	1.27
JH-256	2,661	1,612	8,392	0.25	0.25	0.80	0.80
Keccak(-256)	1,433	2,666	4,806	0.29	0.29	0.29	0.29
Luffa-256	1,048	1,446	3,754	0.24	0.24	0.36	0.43
Shabal-256	1,251	2,061	4,219	0.23	0.23	1.19	2.15
SHAvite-3 <sub>256</sub>	1,063	1,363	3,564	0.24	0.24	0.73	0.73
SIMD-256	3,987	6,693	13,908	0.29	0.29	1.09	1.09
Skein-256-256	854	929	2,864	0.26	0.26	0.86	1.08

# Evaluation Analysis

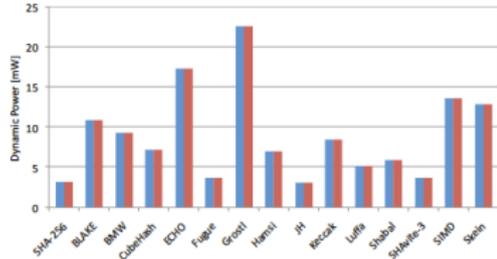
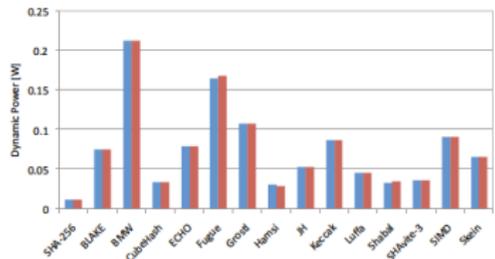
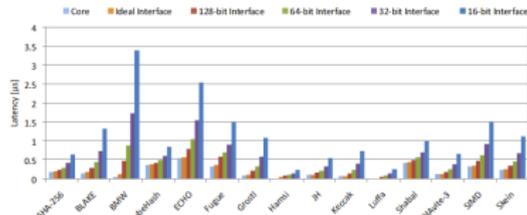
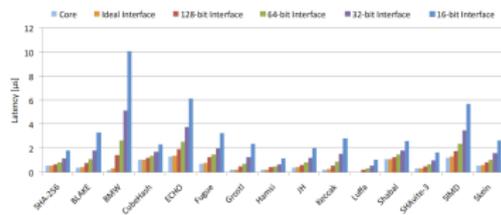
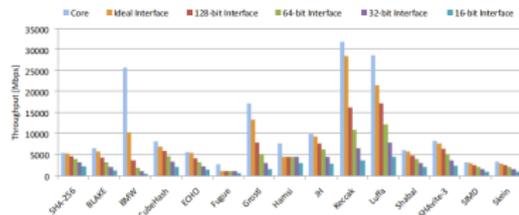
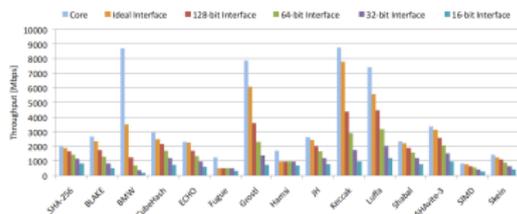
◆ SHA-256 ■ BLAKE ▲ BMW ✕ CubeHash ✕ ECHO ● Fugue + Grøstl - Hamsi - JH ◆ Keccak ■ Luffa ▲ Shabal ✕ SHAvite-3 ✕ SIMD ● Skein



◆ SHA-256 ■ BLAKE ▲ BMW ✕ CubeHash ✕ ECHO ● Fugue + Grøstl - Hamsi - JH ◆ Keccak ■ Luffa ▲ Shabal ✕ SHAvite-3 ✕ SIMD ● Skein



# FPGA vs ASIC



# What We Learned from This Project

- World-wide Collaboration is important for hardware evaluation
  - Giving trust to the evaluation results
  - Share efforts for evaluation
- The collaboration should be open and public
  - Consistent evaluation criteria and public verifiability are important tools for “Fair Evaluation.”
  - Use publicly available evaluation platform
  - Exchange data and source codes over the Internet
  - Open-source like management is much effective

# Conclusion

- Define evaluation criteria for consistent and fair evaluation
  - Platform
  - Interface
  - Evaluation items
- Manage evaluation project by world-wide collaboration
- Show evaluation results on SASEBO-GII FPGA board
  - Speed performance
  - Implementation cost
  - Power consumption